Postal Regulatory Commission
Submitted 8/19/2011 3:59:44 PM
Filing ID: 75006
Accepted 8/19/2011

```
OPTIONS ERRORS=0 DQUOTE MPRINT SYMBOLGEN;

  /* NOTE:  The library and macro variable statements below were amended for the
purposes of this re-work; at one time they pointed to two different directories
*/
%LET uspsdata=C:\Users\kevin.k.gillette\Documents\USPS Retail Self-Service
Project\Additional work 2011-07-21;
LIBNAME uspsdata "&uspsdata.";
%LET rdmdata=&uspsdata.;
LIBNAME rdmdata "&uspsdata.";




  /*************** STEP0:  Import latitude/longitude data as retrieved from the
EDW table, EDWFcltyProdView.FDB_Cmn_Core, as well as the three
                           spreadsheet tabs with Small post offices identified
(originally provided by Tom Ford from USPS) ***************/

PROC IMPORT DATAFILE="&rdmdata.\FDB CMN Core.xlsx"
    OUT=uspsdata.fdb_cmn_core
      REPLACE;
    SHEET="FDB_CMN_CORE";
    GETNAMES=YES;
RUN;

PROC IMPORT DATAFILE="&rdmdata.\FY2010 Offices under $100,000 by Technology.xls"
    OUT=uspsdata.pos_small
      REPLACE;
    SHEET="POS";
    GETNAMES=YES;
RUN;

PROC IMPORT DATAFILE="&rdmdata.\FY2010 Offices under $100,000 by Technology.xls"
    OUT=uspsdata.irt_small
      REPLACE;
    SHEET="IRT";
    GETNAMES=YES;
RUN;

PROC IMPORT DATAFILE="&rdmdata.\FY2010 Offices under $100,000 by Technology.xls"
    OUT=uspsdata.emoves_small
      REPLACE;
    SHEET="Emoves (Manual)";
    GETNAMES=YES;
RUN;

%MACRO keep_ufn(ds);
DATA uspsdata.&ds.;
    SET uspsdata.&ds.;
    ufn = unit_finance_number + 0;    /* Ensure that it's numeric */
    KEEP ufn;
RUN;
%MEND keep_ufn;

%keep_ufn(pos_small);
%keep_ufn(irt_small);
```

```
%keep_ufn(emoves_small);

DATA uspsdata.small_post_offices;
    SET uspsdata.pos_small uspsdata.irt_small uspsdata.emoves_small;
RUN;




  /*************** STEP1:  Convert latitude/longitude data into radians, as
these are necessary for use of SAS trigonometric functions below.  Any post
offices
                           with missing latitude/longitude data will be deleted.
The field labeled "unitfinancenumber2" was built during the Teradata extract,
                           and is the direct concatenation of Chargeable Finance
Number and Unit Number.  This turned out to be a better number than the field
                           that was actually called Unit Finance Number, which
was frequently NULL. *************/

DATA uspsdata.fdb_cmn_core;
    SET uspsdata.fdb_cmn_core;
    IF _N_ = 1 THEN two_pi = 8.0000 * ATAN(1.0000);
    RETAIN two_pi;
    latitude = (latitude * two_pi) / 360.0;
    longitude = (longitude * two_pi) / 360.0;
    IF latitude = . OR longitude = . THEN DELETE;
    ufn = unitfinancenumber2 + 0;     /* The JOINed UFN is more accurate; ensure
that it's numeric */
    IF ufn = . THEN DELETE;
    DROP two_pi unitfinancenumber unitfinancenumber2;
RUN;




  /************** STEP2:  Make two copies of the latitude/longitude data, for
use in the Cartesian product mapping step below (STEP3) ***************/

DATA set1;
    SET uspsdata.fdb_cmn_core;
    lat1 = latitude;
    lon1 = longitude;
    ufn1 = ufn;
    OUTPUT;
    KEEP lat1 lon1 ufn1;
RUN;

DATA set2;
    SET uspsdata.fdb_cmn_core;
    lat2 = latitude;
    lon2 = longitude;
    ufn2 = ufn;
    OUTPUT;
    KEEP lat2 lon2 ufn2;;
RUN;
```

```
  /************** STEP3:  Now we'll run the N(N-1) product (Cartesian, omitting
'diagonal' entries) routine to figure out which neighbors exist, and
                          how close they are.  Since the latitude/longitude data
are occasionally suspect, we omit any 'neighbors' with a distance of less
                          than 1/10-th of a mile, which is 176 yards (=
1760/10).  This sometimes happens when facilities are collocated, or (as alluded
above)
                          one or both of the latitude/longitude data points is
not entirely accurate. ************/

DATA uspsdata.fdb_nearest_neighbors;
    SET set1;
    IF _N_ = 1 THEN rec_cnt = 0;
    RETAIN rec_cnt;
    LENGTH ufn_nearest $ 10;
    ufn_nearest = "";
    dist_nearest = 999999999999;
    DO ptr=1 TO nobs2;
        SET set2 POINT=ptr NOBS=nobs2;
        rec_cnt+1;
        IF MOD(rec_cnt,100000000) = 0 THEN
          PUT "Combinations processed: " rec_cnt;
        IF ufn1 NE ufn2 THEN DO;
              /* This formula is a conventional application of spherical
geometry, obtainable from numerous on-line sources and any high-school level
text on
                 trigonometry.  The accompanying document describes the
calculation in greater detail. */
            delta_phi = lon1 - lon2;
            theta = ARCOS((SIN(lat1)*SIN(lat2)) +
(COS(lat1)*COS(lat2)*COS(delta_phi)));
            dist = 3959.9 * theta;      /* Distance in miles */
            IF dist ^= . THEN DO;
                IF 0.10 <= dist AND dist < dist_nearest THEN DO;      /* 0.10
miles = 176 yards - typically this happens with collocated facilities or bad
lat/long data */
                    dist_nearest = dist;
                    ufn_nearest = LEFT(TRIM(ufn2));
                END;
            END;
        END;
    END;
    unit_finance_number = ufn1;
    unit_finance_number_nearest = ufn_nearest;
    dist_to_nearest_neighbor = dist_nearest;
    OUTPUT;
    KEEP unit_finance_number unit_finance_number_nearest
dist_to_nearest_neighbor;
RUN;
```

```
  /************** STEP4:  Mark those post offices that are considered "Small"
*******************/

PROC SQL;
    CREATE TABLE foo AS
      SELECT
        fnn.*,
        CASE WHEN spo.ufn IS NULL THEN ' ' ELSE 'Y' END    AS pos_is_lt100k
      FROM uspsdata.fdb_nearest_neighbors       fnn
      LEFT JOIN uspsdata.small_post_offices     spo
      ON fnn.unit_finance_number = spo.ufn
    ;
QUIT;

DATA uspsdata.fdb_nearest_neighbors;
    SET foo;
RUN;




  /************** STEP5:  Push the nearest-neighbor data out to Excel for
further analysis *******************/

PROC EXPORT DATA=uspsdata.fdb_nearest_neighbors
    DBMS=EXCEL2000
    OUTFILE="&uspsdata.\Nearest neighbor data.xls"
    REPLACE;
  SHEET="Neighbor data";
RUN;
```